Chapter 1

String Acceptors

1.1 Deterministic Finite-state String Acceptors

1.1.1 Orientation

This section is about deterministic finite-state acceptors for strings. The term *finite-state* means that the memory is bounded by a constant, no matter the size of the input to the machine. The term *deterministic* means there is single course of action the machine follows to compute the output from some input. As we will see later, *non-deterministic machines* can be thought of as pursuing multiple computations simultaneously. The term *acceptor* is synonymous with *recognizer*. It means that this machine solves *membership problems*: given a set of objects X and input object x, does x belong to X? The term *string* means we are considering the membership problem over stringsets. So X is a set of strings (so $X \subseteq \Sigma^*$) and the input x is a string.

1.1.2 Definitions

Definition 1. *A* deterministic finite-state acceptor (DFA) *is a tuple* $(Q, \Sigma, q_0, F, \delta)$ *where*

- Q *is a finite set of states;*
- Σ *is a finite set of symbols (*the alphabet*);*
- $q_0 \in Q$ *is the* initial *state;*
- $F \subseteq Q$ *is a set of* accepting *(*final*) states; and*
- δ *is a function with domain* $Q \times \Sigma$ *and co-domain* Q*. It is called the* transition function.

We extend the domain of the transition function to $Q \times \Sigma^*$ as follows. In these notes, the empty string is denoted with λ.

$$\begin{aligned} \delta^*(q, \lambda) &= q \\ \delta^*(q, aw) &= \delta^*((\delta(q, a), w) \end{aligned} \tag{1.1}$$

Consider some DFA $A = (Q, \Sigma, q_0, F, \delta)$ and string $w \in \Sigma^*$. If $\delta^*(q_0, w) \in F$ then we say A *accepts/recognizes/generates* w. Otherwise A *rejects* w.

Definition 2. *The stringset recognized by* A *is* $L(A) = \{w \in \Sigma^* \mid \delta^*(q_0, w) \in F\}$.

The use of the 'L' denotes "Language" as stringsets are traditionally referred to as *formal languages.*

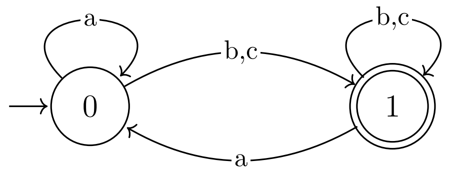
Definition 3. *A stringset is* regular *if there is a DFA that recognizes it.*

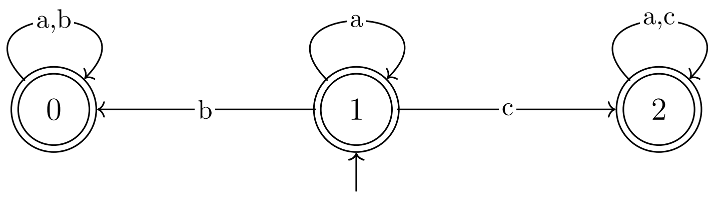
1.1.3 Exercises

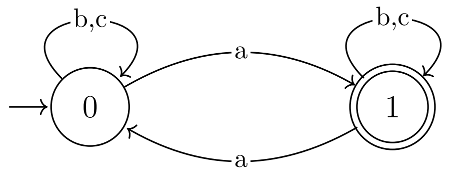
Exercise 1. This exercise is about designing DFA. Let $\Sigma = \{a, b, c\}$. Write DFA which express the following generalizations on word well-formedness.

1. All words begin with a consonant, end with a vowel, and alternate consonants and vowels.
2. Words do not contain aaa as a substring.
3. If a word begins with a, it must end with c.
4. Words must contain two bs.
5. All words have an even number of vowels.

Exercise 2. This exercise is about reading and interpreting DFA. Provide generalizations in English prose which accurately describe the stringset these DFA describe.

1. 

2. 

3. 

4. Write the DFA in #1-3 in mathematical notation. So what is Q, Σ, q_0, F, and δ?

1.2 Properties of DFA

Note that for a DFA A, its transition function δ may be partial. That is, there may be some $q \in Q, a \in \Sigma$ such that $\delta(q,a)$ is undefined. If δ is a partial function, δ^* will be also. It is assumed that if $\delta^*(q_0,w)$ is undefined, then A rejects w.

We can always make δ total by adding one more state to Q. To see how, call this new state $\diamondsuit$. Then for each $(q,a) \in Q \times \Sigma$ such that $\delta(q,a)$ is undefined, define $\delta(q,a)$ to equal $\diamondsuit$. Every string which was formerly undefined w.r.t. to δ^* is now mapped to $\diamondsuit$, a non-accepting state. This state is sometimes called the *sink* state or the *dead* state.

Definition 4. *A DFA is* complete *if δ is a total function. Otherwise it is* incomplete*.*

It is possible to write DFA which have many useless states. A state can be useless in two ways. First, there may be no string which forces the machine to transition into the state. Second, there may be a state from which no string

Definition 5. *A state q in a DFA A is* useful *if there is a string w such that $\delta^*(q_0,w) = q$ and a string v such that $\delta^*(q,v) \in F$. Otherwise q is* useless*. If every state in A is useful, then A is called* trim*.*

Not all complete DFAs are trim. If there is a sink state, it is useless in the above sense of the word.

Definition 6. *A DFA A is* minimal *if there is no other DFA A' such that $L(A) = L(A')$ and A' has fewer states than A.*

Technically, not all complete DFAs are minimal. If there is a sink state, it is not minimal.

Exercise 3. Consider the DFAs in the exercise 2. Are they complete? Trim? Minimal?

1.3 Some Closure Properties of Regular Languages

A set of objects X is *closed* under an operation $\circ$ if for all objects $x, y \in X$ it is the case that $x \circ y \in X$ too.

We can easily show that the union of any two regular stringsets R and S is also regular. Let $A_R = (Q_R, \Sigma, q_{0R}, F_R, \delta_R)$ be the DFA recognizing R and let $A_S = (Q_S, \Sigma, q_{0S}, F_S, \delta_S)$ be the DFA recognizing S. We can assume A_R and A_S are complete. We assume the same alphabet.

Construct $A = (Q, \Sigma, q_0, F, \delta)$ as follows.

- $Q = Q_R \times Q_S$.
- $q_0 = (q_{0R}, q_{0S})$.
- $F = \{(q_r, q_s) \mid q_r \in F_R \textbf{ or } q_s \in F_S\}$.
- $\delta((q_r, q_s), a) = (q'_r, q'_s)$ where $\delta_R(q_r, a) = q'_r$ and $\delta_S(q_s, a) = q'_s$.

Theorem 1. *$L(A) = R \cup S$.*

Similarly, the same kind of construction shows that the intersection of any two regular stringsets is regular. Construct $B = (Q, \Sigma, q_0, F, \delta)$ as follows.

- $Q = Q_R \times Q_S$.
- $q_0 = (q_{0R}, q_{0S})$.
- $F = \{(q_r, q_s) \mid q_r \in F_R$ **and** $q_s \in F_S\}$.
- $\delta((q_r, q_s), a) = (q_r', q_s')$ where $\delta_R(q_r, a) = q_r'$ and $\delta_S(q_s, a) = q_s'$.

Theorem 2. *$L(B) = R \cap S$.*

Here are some additional questions we are interested in for regular stringsets R and S.

1. Is the complement of R (denoted $\overline{R}$) a regular stringset?
2. Is $R \backslash S$ a regular stringset?
3. Can we decide whether $R \subseteq S$?
4. Is RS a regular stringset? (Note $RS = \{rs \mid r \in R \text{ and } s \in S\}$
5. Is R^* a regular stringset? (Note $R^0 = \{\lambda\}$, $R^n = R^{n-1}R$, $R^* = \bigcup_{n \in \mathbb{N}^0} R^n$)

The answers to all of these questions is Yes. With a little thought about complete DFA, the answers to first three follow very easily.

Theorem 3. *If R is a regular stringset then the complement of R is regular.*

Proof (Sketch). If R is a regular stringset then there is a complete DFA $A = (Q, \Sigma, q_0, F, \delta)$ which recognizes it. Let $B = (Q, \Sigma, q_0, F', \delta)$ where $F' = Q \backslash F$. We claim $L(B) = \overline{R}$. □

Corollary 1. *If R, S are regular stringsets then so is $R \backslash S$ since $R \backslash S = R \cap \overline{S}$.*

Corollary 2. *If R, S are regular stringsets then it is decidable whether $R \subseteq S$ since $R \subseteq S$ iff $R \backslash S = \varnothing$.*

Corollary 3. *If R, S are regular stringsets then it is decidable if $R = S$ since $R = S$ iff $R \subseteq S$ and $S \subseteq R$.*

We postpone explaining how and why for the last two questions.